

## Section 2: Linear Transverse Motion and Toy Lattices

### □ Transverse application

- directory structure

### □ User-oriented interface

- `./src/run.cc`: user's main program

### □ ADXF lattice file

- `eq_tune_fodo.adxf`
- accelerator elements and sectors

### □ Simulation

- demo
- examples

# Directory structure

---

- **Makefile**
- **src**: source code **run.cc** of user's main program and various trackers.
- **apdf**: tracker files
- **lib**: temporary object files
- **linux**: executable program **run** generated from the **src** files

Our primary interest in  
this section

To recompile this application:

**make clean**

**make**

To start this application:

**./linux/run ring ..\lattices\eq\_tune\_fodo.adxf ./apdf\teapot.apdf**

## run.cc

---

```
// *****
std::cout << "Read the ADXF file (lattice description)." << std::endl;
// *****

if(latticeFile.find(".adx") != std::string::npos) {
    shell.readADXF(Args() << Arg("file", latticeFile));
} else {
    shell.readSXF(Args() << Arg("file", latticeFile));
}

// *****
std::cout << "Select a lattice." << std::endl;
// *****

shell.use(Args() << Arg("lattice", lattice));

// *****
std::cout << "Set beam attributes." << std::endl;
// *****

double mass  = 183.43261;
double energy = 23.75*mass; // 109*mass

shell.setBeamAttributes(Args << Arg("charge", 79) << Arg("mass", mass) << Arg("energy", energy));
```

**user-oriented commands**

**comments**

## run.cc (cont)

---

```
// ****
std::cout << "Read the APDF file (propagator description)." << std::endl;
// ****

shell.readAPDF(Args() << Arg("file", apdfFile ));

// ****
std::cout << "Generate a bunch distribution." << std::endl;
// ****

shell.setBunch(Args()
    << Arg("np", 1)           // number of particles
    << Arg("enx", 15.0e-6)    // normalized horizontal emittance [pi*m*rad]
    << Arg("eny", 15.0e-6)    // normalized vertical emittance [pi*m*rad]
    << Arg("ctMax", 0)        // half bunch length [m]
    << Arg("deMax", 0)        // half relative energy spread
    << Arg("seed", -100));    // seed number

player->setTurns(1000); // number of turns
player->setFprint(250); // frequency of printing. [update/turn]
```

# Accelerator Description Exchange Format (ADXF 2.x)

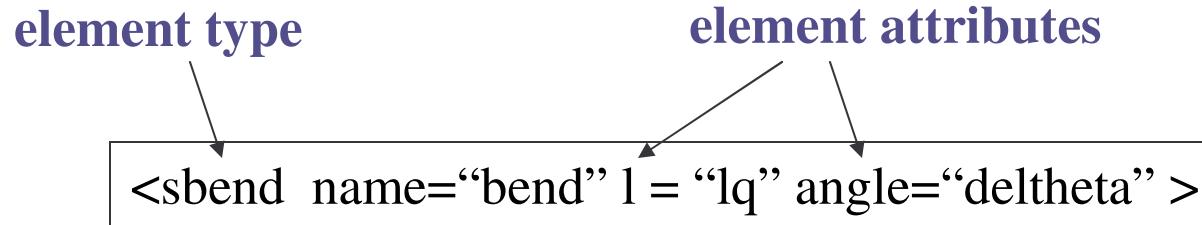
## eq\_tune\_fodo.adxf

```
<?xml version="1.0" encoding="utf-8"?>
<adx> xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance xsi:noNamespaceSchemaLocation="file:../../doc/adxf/adxf.xsd">
  <constants>
    <!-- if nhalf is changed, # of cells must be changed in "lattice" -->
    1   <constant name="pi"  value="3.1415926536" />
    <constant name="twopi" value="2.0*pi" />
    ...
  </constants>
  <!-- define magnetic elements -->
  2   <elements>
    <marker name="mbegin" />
    ...
    <sbend name="bend" l="l0" angle="deltheta" />
    <quadrupole name="quadhf" l="lq" k1="kq1" />
    <quadrupole name="quadvf" l="lq" k1="kq2" />
    <sextupole name="sext1" l="ls" k2="ks1" />
    <sextupole name="sext2" l="ls" k2="ks2" />
    <monitor name="bpm" />
    <kicker name="kicker" />
  </elements>
  <sectors>
    3   <sector name="fullcell" line="mbegin bpm quadhf sext1 bpm bend sext2 quadvf
          bpm quadvf sext2 bpm bend sext1 quadhf mbegin" />
    <sector name="ring" line = "mbegin kicker fullcell fullcell fullcell fullcell
          fullcell fullcell fullcell fullcell fullcel mend" />
  </sectors>
</adx>
```

# ADXF Lattice File

## Accelerator elements

---



### MAD accelerator elements:

1. **Marker:** <marker name="mk1" >
  - no attributes
2. **Sector bending magnet:** <sbend name="bend" l="lq" angle="deltheta" />
  - l: length [m]
  - angle: bend angle [rad]
3. **Quadrupole:** <quadrupole name="quadhf" l="lq" k1="kq1" />
  - l: length [m]
  - k1: strength,  $\frac{1}{B\rho} \frac{\partial B_y}{\partial x}$  [m<sup>-2</sup>]
4. **Sextupole:** <sextupole name="sext1" l="ls" k2="ks1" />
  - l: length [m]
  - k2: strength,  $\frac{1}{B\rho} \frac{\partial^2 B_y}{\partial x^2}$  [m<sup>-3</sup>]

# ADXF Lattice File (cont)

## Accelerator elements and sectors

---

### MAD accelerator elements (cont):

#### 5. Kicker: <kicker name="kicker1" />

- l: length [m]
- hkick: horizontal kick angle [rad]
- vkick: vertical kick angle [rad]

#### 6. Monitor: <monitor name="bpm" />

- l: length [m]

### Sectors:

sequence of elements and/or sectors

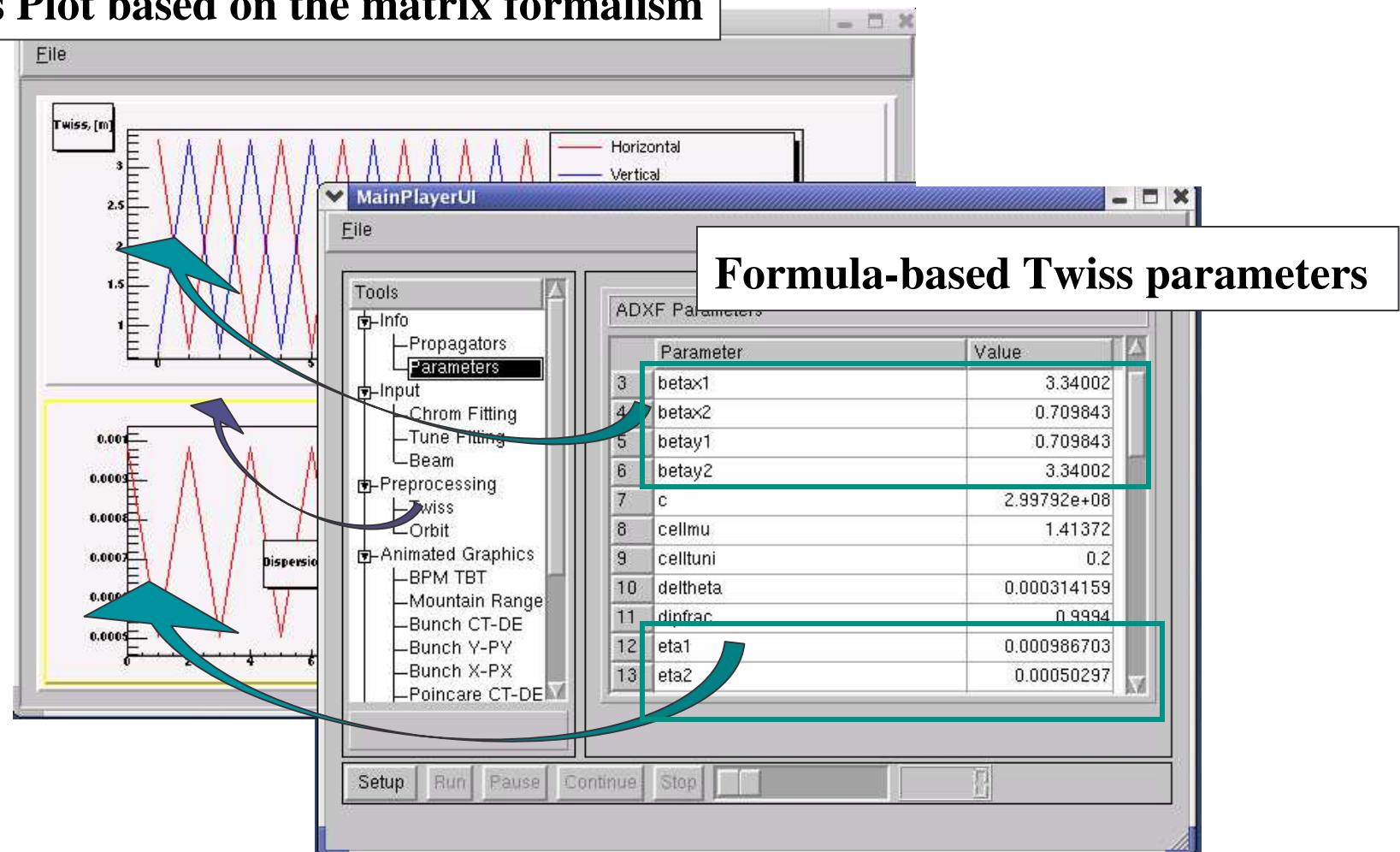


```
<sector name="fullcell" line = "mk1  
                      bpm quadhf sext1 bpm bend sext2 quadvf  
                      bpm quadvf sext2 bpm bend sext1 quadhf  
                      mk1" />
```

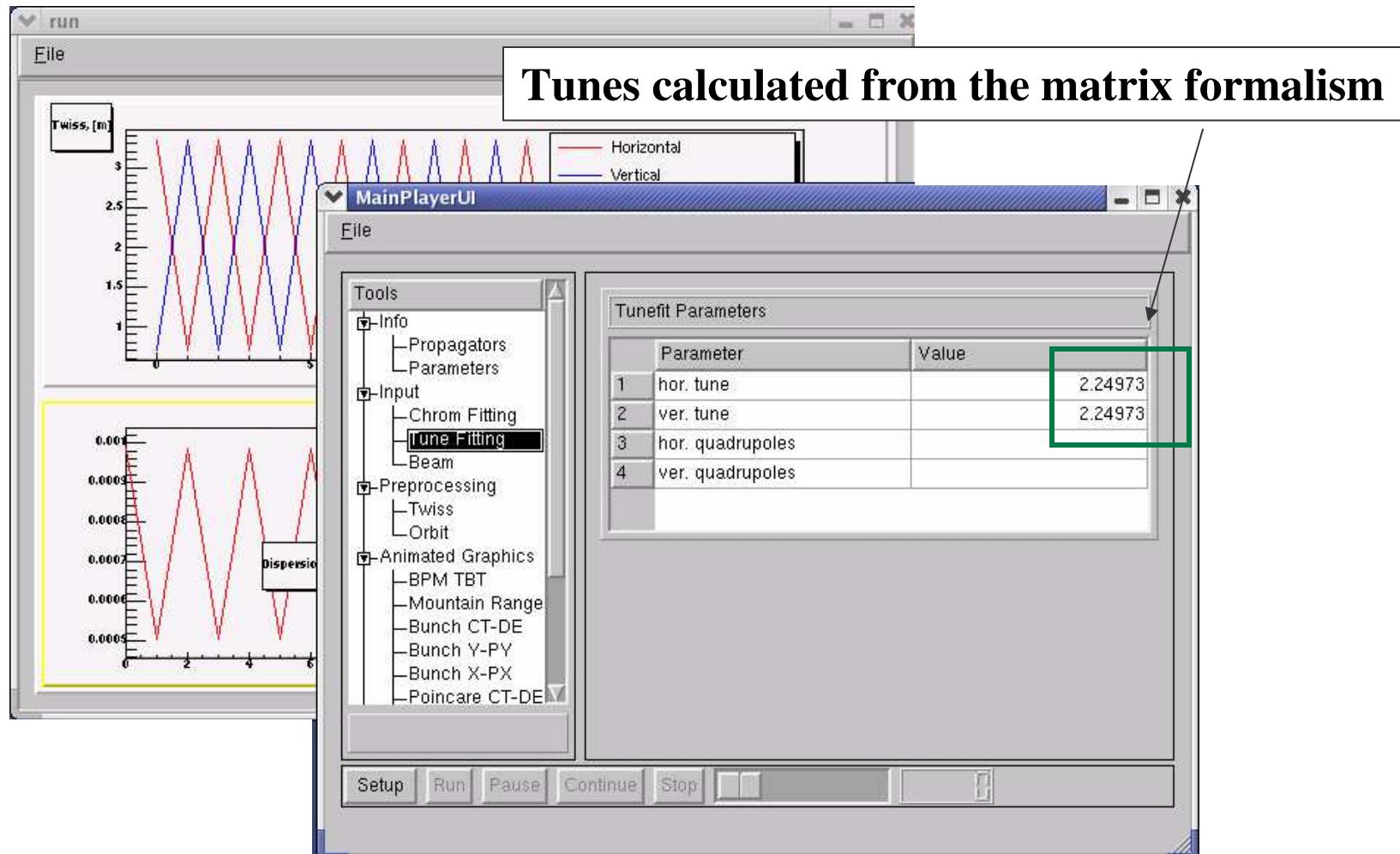
# Transverse Application

`./linux/run ring ./lattices/eq_tune_fodo.adxf ./apdf/teapot.apdf`

## Twiss Plot based on the matrix formalism



# TuneFit Editor



# Simulation Examples

---

- **S 3.1:** change `nufrac` in `eq_fodo_tune.adxf` and check/record the modified values from the Parameter Browser.
- **S 3.2:** compare formula-based lattice parameters (`betax1`, `eta1`, etc) in the Parameter Browser with the Twiss Viewer plots calculated from the matrix formalism
- **S 3.3:** change `deltheta` to minimize the effect of dipole focusing and improve the agreement in the previous example.
- **S 3.4:** repeat S 3.1 for the `general_fodo.adxf` lattice
- **S 3.5:** evaluate  $dv/dq$  from S 3.4 modified values and compare it with Eq. 3.47
- **S 3.6:** see the effect of “the golden rule” (Eq. 3.52) by replacing quadrupole strengths with  $q1t/lq$  and  $q2t/lq$  values.